UNIVERSIDAD TECNOLÓGICA METROPOLITANA
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INFORMÁTICA Y COMPUTACIÓN
ESCUELA DE INFORMÁTICA

Profesores: Mauro Castillo Valdés
Ayudantes: Annabella Carvajal M.
Melany Farias S.

PRUEBA Nº 1 * INF 628 * ESTRUCTURAS DE DATOS * SEM 2-2010

Nombre:_____

Preguntas 1-6 : 4*BUENAS – MALAS
Pregunta 7 : 8 puntos
Pregunta 8 : 9 puntos
Pregunta 9 : 19 puntos

1. Indique el valor que se imprime:

```
int m=4;
float r=9,5;
int funcion(int a, int *b) { return a * (*b);}
int funcion(float c, int *d) { return c + (*d);}
int funcion(doble e, int *f) { return e / (*f);}
cout << funcion(r, &m) << endl;
```

a) 7
b) 3.0
c) 0
d) Ninguna de las anteriores **RESPUESTA**

2. Al ejecutar el siguiente código:

```
char *t = new char[3];
*t = 'a'; t++;
*t = 'b'; t++;
*t = 'c'; t--;
 cout << *t;
```

a) a
b) b **RESPUESTA**
c) c
d) Ninguna de las anteriores

3. Al definir la siguiente función usando plantillas:

```
template <class T, class R>
double ver(T x, R y)
{
  return x / y;
}
```

Son correctas?

I. cout << ver(3,6.4) << endl;
II. cout << ver(6.4,3) << endl;
III. cout << ver(6.4,3.3) << endl;

a) Sólo I y II
b) Sólo II y III
c) todas **RESPUESTA**
d) Ninguna de las anteriores

4**. En base a la declaración de la clase Temporal:**

```
class temporal{
  private:
        int aux;
        int doble(){return aux*2;}
  public:
        temporal(){aux=5;}
        int triple(){return aux*3;}
};
```

Indique cuál sentencia es correcta:

a) temporal T = 6;
b) cout << T.doble() << endl;
c) cout << T.triple() << endl; **RESPUESTA**
d) Ninguna de las anteriores

5. En base a la clase principal, indique que se imprime:

```
class principal{
  private:
        int v;
  public:
        principal(){v=0;}
        principal(int x){v=x;}
        principal(double x){v=(int)x;}
        principal(principal &C){v=C.v;}
        void mostrar(){cout << v << endl;}
};
int main()
{  principal P;
   principal R(3.4);
   principal T(R);
   P.mostrar();
   R.mostrar();
   T.mostrar();
}
```

a) 3 3 0
b) 0 3 3 **RESPUESTA**
c) 0 4 4
d) Ninguna de las anteriores

6. Según la definición:

```
class caja{
  private:
        int pro;
  public:
        caja( ){pro=0;}
        void set(int v){pro=v;}
        int get( ){return pro;}
};

caja *c = new caja;
```

La instrucción incorrecta para poder acceder al objeto:
a) c->set(3);
b) cout << c->get() << endl;
c) cout << (*c).get() << endl;
d) Ninguna de las anteriores **RESPUESTA**

7. En base a la clase Cuadrado:

```
template <class T>
class Cuadrado{
    private:
            T lado;
    public:
            Cuadrado( );
            Cuadrado(T x);
            void setLado(T x);
            T getLado( );
            void mostrar( );
            T area( );
};
```

a) Escribir los métodos de la clase Cuadrado (4 puntos)

RESPUESTA:

```
template <class T>
Cuadrado<T>::Cuadrado()
{
  lado = 0;
}
template <class T>
Cuadrado<T>::Cuadrado(T x)
{
  lado = x;
}
template <class T>
void Cuadrado<T>::setLado(T x)
{
  lado = x;
}
template <class T>
T Cuadrado<T>::getLado()
{
  return lado;
}
template <class T>
void Cuadrado<T>::mostrar()
{
  cout << "Cuadrado con lado : " << lado << endl;
}
template <class T>
T Cuadrado<T>::area()
{
  return lado*lado;
}
```

b) Escribir la función externa Perimetro(C) (4 puntos)

RESPUESTA:

```
template <class T>
T Perimetro(Cuadrado<T> C)
{
  return 4*C.getLado();
}
```

8. Con respecto a la clase Punto y Cuadrilátero

```
class Punto {
  private:
    double x;
    double y;
  public:
    Punto(double a=0, double b=0);
    void setX(double a);
    void setY(double b);
    double getX( );
    double getY( );
    void mostrar( );
};
```

```
class Cuadrilatero {
  private:
    Punto V[4]; //definido con 4 puntos

  public:
    Cuadrilatero(double, double, double, double,
                 double, double, double, double);
    Punto get1( ){return V[0];}
    Punto get2( ){return V[1];}
    Punto get3( ){return V[2];}
    Punto get4( ){return V[3];}

    void mostrar( );
};
```

```
Int main( ){
   Punto T(0,0);
   Punto R(3,3);
   double d = Distancia(T, R);
   cout << "distancia entre T y R es: " << d << endl;

   Cuadrilatero C(1,1,3,2,4,4,2,3);
   double p = Perimetro(C);
   cout << "Perimetro es : " << p << endl;
}
```

Se pide escribir las funciones externas:
a) Distancia(P1, P2) : retorna la distancia entre los puntos P1 y P2 (4 puntos)
RESPUESTA:

```
double distancia(Punto R, Punto T)
{
  return sqrt((R.getX() - T.getX())*(R.getX() - T.getX()) +
          (R.getY() - T.getY())*(R.getY() - T.getY()));
}
```

b) Perimetro(C) : retorna el perímetro del cuadrilátero C (5 puntos)

RESPUESTA:

```
double Perimetro(Cuadrilatero W)
{
   double d=0.0;
   d = d + distancia(W.get1(), W.get2());
   d = d + distancia(W.get2(), W.get3());
   d = d + distancia(W.get3(), W.get4());
   d = d + distancia(W.get4(), W.get1());
   return d;
}
```

9. Definir una clase CONJUNTO que contenga:

- **Un array de valores de cualquier tipo**
- **El array tiene una cantidad máxima de 20 elementos**
- **Un método Intersección que tome como argumento otro conjunto, y devuelva un nuevo conjunto con la intersección de los dos, es decir, los elementos del primer conjunto que son iguales al algún elemento del segundo.**

a) Definir la clase con sus datos privados (4 puntos)
RESPUESTA

```
template <class T>
class Conjunto{
    private:
      T v[MAX];
      int largo;
    public:
          Conjunto();
          void Mostrar();
          void Agregar(T);
          bool Pertenece(T);
          int getLargo();
          Conjunto Interseccion(Conjunto &C);
};
```

b) Definir cada uno de sus métodos utilizados (5 puntos)
RESPUESTA:

```
template <class T>
Conjunto<T>::Conjunto()
{
  largo = 0;
}

template <class T>
void Conjunto<T>::Mostrar()
{
  for(int i=0; i<largo; i++)
    cout << "Elemento : " << v[i] << endl;
}
template <class T>
void Conjunto<T>::Agregar(T e)
{
    if(!Pertenece(e))
      v[largo++] = e;
    else
      cout << "El elemento " << e << " ya se encuentra en el conjunto " << endl;
}
template <class T>
bool Conjunto<T>::Pertenece(T e)
{
  bool esta = false;
  for(int i=0; i<largo; i++)
    if(v[i]==e)
      esta=true;
  return esta;
}
template <class T>
int Conjunto<T>::getLargo()
{
  return largo;
}
template <class T>
```

c) Declarar el método Interseccion(Conjunto C) (10 puntos)
RESPUESTA:

```
Conjunto<T> Conjunto<T>::Interseccion(Conjunto &C)
{
   Conjunto<T> AUX;
   for(int i=0; i<largo; i++)
   {
     if(C.Pertenece(v[i]))
       AUX.Agregar(v[i]);
   }
   return AUX;
}
```